Università degli Studi di Bologna

Corso di Laurea in Informatica
Esame scritto di LOGICA PER L'INFORMATICA (9/6 CFU)
14/07/2022

Scrivere **nome, cognome, numero di CFU e numero di matricola in alto a destra** in tutti i fogli protocollo. Gli esercizi con un doppio punteggio riportano prima il punteggio nel caso dell'esame da 9 CFU e poi quello nel caso di esame da 6 CFU. Fare attenzione all'esercizio 10 che è diverso a seconda del numero di CFU.

1 (1 punto). Dare la sintassi per le formule della logica del prim'ordine.

2 (4 punti/6 punti). Considerare le seguenti grammatiche per liste di elementi generati da un non-terminale X e per alberi binari i cui nodi sono tutti etichettati con numeri naturali:

$$\begin{array}{lll} L & ::= & [] \mid X :: L \\ T & ::= & < \mathbb{N} > \mid T * \mathbb{N} * T \end{array}$$

Scrivere, per ricorsione strutturale, una funzione $c(T)$ che, **dato in input un albero T, restituisca la lista di tutti i suoi cammini foglia-radice** dove un cammino foglia-radice è la lista di numeri naturali incontrati risalendo da una foglia dell'albero verso la radice.

Esempio:

```
c((<2>*3*<1>)*8*(<6>*4*(<1>*2*<2>))) =
 (2::3::8::[]) ::
 (1::3::8::[]) ::
 (6::4::8::[]) ::
 (1::2::4::8::[]) ::
 (2::2::4::8::[]) ::
 []
```

È possibile utilizzare funzioni ausiliare e/o parametri ausiliari. Nel caso di uso di parametri ausiliari per la funzione principale f, specificare il valore iniziale da passare per risolvere il problema.
Dettagliare la specifica di tutte le funzioni ausiliarie.
Testare ogni funzione implementata su un input di esempio.

3 (5 punti). Dimostrare in teoria assiomatica degli insiemi che

$$\forall A \forall B \forall C((A \cup B) \cap B = (C \cap B) \cup B)$$

Prima della dimostrazione riportare l'**enunciato di tutti gli assiomi di teoria degli insiemi che usate nella dimostrazione**.

La dimostrazione deve essere una dimostrazione valida in logica del prim'ordine, ovvero ogni passaggio deve corrispondere a uno o più passaggi di deduzione naturale classica o intuizionista. Preferire una prova intuizionista ove possibile.

4 (1 punto). Dare la definizione di insieme infinito.

5 (1 punto). Dare la definizione di regola invertibile in deduzione naturale.

6 (1 punto). $\forall x(P(x) \vee Q(x)) \iff (\forall x P(x)) \vee (\forall x Q(x))$: vero o falso? Giustificare la risposta.

7 (5 punti). Considerare la seguente funzione che elimina duplicati consecutivi da una lista:

```
uniq [] = []
uniq (x :: l) = if isHead x l then uniq l else x :: uniq l
```

dove

```
isHead x [] = false
isHead x (y :: l) = x == y
```

Dimostrare, per induzione strutturale su l, che

(a) per ogni x, l si ha che se `isHead x l = true` allora per ogni y si ha `isHead y l = true` sse $x = y$

(b) per ogni x, l si ha `isHead x l = isHead x (uniq l)`

Suggerimento: nella prova di (b) è ovviamente possibile utilizzare (a) come lemma.

8 (6 punti). Si consideri il seguente ragionamento:

Il contratto è in nero o non si fanno straordinari e si prende la quattordicesima. Il contratto non è in nero e non c'è la quattordicesima o

si fanno straordinari. Quindi, se non ci sono alternative, si fanno gli straordinari con un contratto in nero.

Verificare la correttezza del ragionamento utilizzando la deduzione naturale per la logica proposizionale. Preferire una prova intuizionista se possibile.

9 (1 punto/2 punti). Effettuare la seguente sostituzione minimizzando il numero di cambi di nome alle variabili.

$$(\forall x.\exists y.\lim_{x \to z+y} x^2 + a = 0)[\mathrm{x}{+}1/a]$$

10 (2 punti). **PER CHI SOSTIENE L'ESAME DA 6 CFU**
Dimostrare il seguente teorema usando la deduzione naturale al prim'ordine, preferendo una prova intuizionista a una classica ove possibile. Considerare la somma associativa a destra.

$$(\forall x.\exists y.x \geq -k + y + k) \Rightarrow \forall x.\exists y.k + x \geq -k + y$$

10 (5 punti). **PER CHI SOSTIENE L'ESAME DA 9 CFU**
Si consideri la seguente dichiarazione di type class `OrderedMonoid` che predica di un tipo `C` (chiamato **sostegno**) l'esistenza di una relazione d'ordine su `C`, di un'operazione binaria su `C` e di un elemento di `C` soggetti a vincoli ulteriori espressi come commenti della type-class:

```
class OrderedMonoid C where
   add :: C -> C -> C     -- add associative
   zero :: C              -- zero neutral element for add
   leq :: C -> C -> bool  -- leq an order relation
                          -- leq x x' /\ leq y y' => leq (add x y) (add x' y')
```

Fornire le seguenti istanze della type-class come richiesto. Nota: istanziare la type-class significa fornire l'**implementazione del codice** e le **prove delle proprietà richieste**.

(a) `OrderedMonoid` $\mathbb{Z}$ dove `add` e `leq` sono le usuali addizione e confronto fra numeri interi

(b) `OrderedMonoid [U]` dove `[U]` è il tipo delle liste di elementi di tipo `U`, `add` concatena le liste e `leq` ne confronta la lunghezza

(c) `OrderedMonoid (U,V)` dove `(U,V)` sono coppie formate da un elemento di tipo `U` e da un elemento di tipo `V`, entrambi monoidi ordinati; l'addizione avviene componente per componente e una coppia è minore o uguale a un'altra coppia se entrambi le rispettive componenti lo sono

Cosa restituisce il seguente frammento di codice? (motivare la risposta)

(d)
```
if leq (3,[1,1,1]) (4,[5,5])
 then add (4,(5,6)) (1,(2,3))
 else add ([[1,2],[3,4]]) ([[4,5]])
```